Problem 3.13: Show that a language is decidable if and only if some enumerator enumerates the language in the standard string order. (Standard string order means that strings are ordered such that a shorter string comes before a longer string in the order.)

Proof:

We first show that if we have a Turing machine M which decides A, then we can create an enumerator E which enumerates A in the standard string order. Let $s_1, s_2, s_3, \ldots$ be the list of all strings in Σ^* given in order.
We give the following high-level description of E:

$E =$ "Ignore the input.

1. Repeat the following for $i = 1, 2, 3, \ldots$
2. Run M on s_i.
3. If M accepts, print s_i."

Now E will print all strings which are accepted by M, and only strings which are accepted by M, and will print them in the standard string order. Therefore, E will enumerate A in the standard string order.

We now show that if we have an enumerator E which enumerates A in the standard string order, then we can create a Turing machine M which decides A.

If A is finite, it is decidable. So we only need to check the case where A is infinite, and E will subsequently never halt. This means that for any string w in A, there will always be infinitely many strings which will be printed out after w which are longer than w, and thus come after w in the standard string order.

The idea of this proof is that if we see a string which is longer than w, we will never see w printed by the enumerator, and then we reject.
We give the following high-level description of M:

$M =$ "On input w:

1. Run E. Every time E outputs a string, compare it to w.
2. If w ever appears in the output of E, *accept*.
3. If E ever outputs a string which is longer than w, *reject*."

Now M will accept all strings printed by E, and only those strings printed by E, so the language decided by M is A.